

## Zachman on the Framework

By John A. Zachman

A number of questions and misconceptions exist about the Framework. I hope my responses will assist in addressing these questions, and clearing up the misconceptions. The following are a summary of the more broad issues of late.

### QUESTIONS:

Does the Zachman framework apply only to design artifacts produced during the software development process, or do "design artifacts" apply to any and all architecture and engineering artifacts produced in the enterprise, including operational systems?

Who uses the Framework? Enterprise Architects, Software Architects, or both? That is, who creates the design artifacts and to what purpose?

### MY RESPONSE:

The Framework is a classification structure for descriptive representations of an object, any object. Since I am interested in Enterprises, I have enterprise names on the descriptive representations. The descriptive representations can be used by anyone or everyone who is interested in the Enterprise and who is authorized to have access to the descriptions. The creators of the artifacts are those who are "subject matter experts" relative to the artifact being created, probably limited to those experts who are authorized to create them and, as assisted by experts trained in the concept transcription and design techniques. For example, if I was the Owner of a building that I wanted built, I would define what I had in mind as to the building (Rows 1 and 2) but I would hire an Architect to transcribe it so I could recognize and approve it and, since it was being produced by an Architect, I would have a warm feeling that it was structurally sound for use by other participants in the process.

Furthermore, the Framework is neutral relative to Organization structure, that is, if I was the CEO and I wanted to write some programs (Column 2, Row 5), I would write some programs. On the other hand, if I was a programmer, I may be constrained to only writing programs (Column 2, Row 5) based on the criteria that are found in the cell above (Column 2, Row 4) and constrained to reusing the data element specifications found in Column 1, Row 5.

Regarding the purpose of the artifacts...

The purpose of the Row 1 artifacts are to define the boundaries of the Enterprise, what is being included (and conceivably, what is being excluded, should that be relevant or necessary).

The purpose of Row 2 artifacts is to conceptually define what the Enterprise "Owners" have in mind.

The purpose of Row 3 artifacts is to design how the Enterprise concepts will be realized systematically, quite independently of any technologies.

The purpose of the Row 4 artifacts is to define the Enterprise implementation based on the general technology constraints being employed.

The purpose of Row 5 artifacts is to specify the implementations to specific technology products being used for implementation

Whether the descriptive representations are used for software or non-software implementations is dependent upon the technology constraints applied in the Row 4 models. For example, if the technologies being employed are pencils, paper and file cabinets, the implementation will likely be called a manual system.

#### QUOTING FROM MY ARTICLE "THE FRAMEWORK FOR ENTERPRISE ARCHITECTURE: Background, Description and Utility"

"The Framework as it applies to Enterprises is simply a logical structure for classifying and organizing the descriptive representations of an Enterprise that are significant to the management of the Enterprise as well as to the development of the Enterprise's systems. It was derived from analogous structures that are found in the older disciplines of Architecture/Construction and Engineering/Manufacturing that classify and organize the design artifacts created over the process of designing and producing complex physical products (e.g. buildings or airplanes.)"

"The older disciplines of Architecture and Manufacturing have accumulated considerable bodies of product knowledge through disciplined management of the "product definition" design artifacts. This has enabled enormous increases in product sophistication and the ability to manage high rates of product change over time. Similarly, disciplined production and management of "Enterprise definition" (e.g. the set of models identified in the Framework for Enterprise Architecture) should provide for an accumulation of a body of Enterprise knowledge to facilitate enormous increases in Enterprise sophistication and accommodation of high rates of Enterprise change over time."

AND THEN ASKING FOR MY COMMENTS ON THE FOLLOWING:

an Enterprise (System) is defined (specified) by the complete set of models.

these sets are classified and then organized in the Framework.

Enterprise Subsystems (e.g. Business Rule Systems, Ontological Systems, Financial Systems, Middleware Systems, Process/Workflow Systems, etc. etc.) are defined by a subset of these models, and these subsystems define, in part, the Enterprise System.

these models can represent the current state of the Enterprise on the one hand, and the future state on the other given that a subset can be parsed for development, and new models can be created.

new development can utilize the classification and organization scheme as well as reuse already existing models. Likewise, there may be a subset that represents a future state of the enterprise without regard to development.

the models are as much about the operational side of the enterprise as the development side.

Do these statements accurately reflect the intent for the use of the Framework? To the degree that they do not, are they consistent with the intent? In other words, would application of the Framework based on the above, violate the intent for use of the Framework or principles of Enterprise Architecture?

HERE ARE MY RESPONSES:

- an Enterprise (System) is defined (specified) by the complete set of models.

An Enterprise system could be manual or automated. In either case, the system creators may or may not build any models. Any models that are defined by the Framework that are not made explicit are implicit ... that is, the system creators are making assumptions about them. In fact, any "slivers" (portions) of models that are not made explicit are implicit ... that is, assumptions are being made.

- these sets are classified and then organized in the Framework.

The Framework is simply a classification scheme ... like the periodic table, it hypothesizes the existence of all of the primitive elements. Any one primitive element that is not being made explicit is just not being made explicit. It is implicit. Also, any model that is produced can be classified by the Framework. Since historically, we were driven to implementation as the principle value proposition, we didn't have time or money to produce primitive models from

which to create the composite models we needed for implementation. Therefore, most models that have been produced over the last 50 years tend to be composite models. That is, the compounds existed before we understood the periodic table. You need the periodic table to turn the alchemy of systems development into a science of Enterprise Architecture (chemistry).

- Enterprise Subsystems (e.g. Business Rule Systems, Ontological Systems, Financial Systems, Middleware Systems, Process/Workflow Systems, etc. etc.) are defined by a subset of these models, and these subsystems define, in part, the Enterprise System.

The "Enterprise Subsystems" (above) are implementations. The question is, were they created from primitive, architectural models or were they simply created as composite models ad hoc for a specific implementation. If they were created ad hoc for a specific implementation, the probability of their being reused is low to zero. If they were created from primitives that were defined relative to the Enterprise (not relative to a specific implementation), the probability of assembling many compounds from the same set of primitives is very high. In this case, you would be "mass customizing" the Enterprise.

- these models can represent the current state of the Enterprise on the one hand, and the future state on the other given that a subset can be parsed for development, and new models can be created.

It is conceivable that you could have an "as is" Framework of primitives and a "to be" Framework of primitives ... It is more likely that you have an "as is" set of horizontal relationships between the primitives and a "to be" set of horizontal relationship between the primitives. It is my opinion that if you define the primitives relative to the Enterprise, they likely do not change appreciably as long as you stay in the same business.

If you get out of airplane manufacturing and get into massage parlors or something, then your primitive models are going to change. As long as you stay in airplane manufacturing, the primitives probably won't change appreciably. Now, having said that, nothing is exempt from change ... and clearly, as your Enterprise Architects gain experience, they are likely to learn more and improve their primitive modeling ability and therefore, their primitive models. If they change the primitive models however, they will likely have some scrap and rework on their hands. That scrap and rework will be minimal as compared to the scrap and rework they would have if they only had composite models. In the composite case, they are into throwing the whole thing out and starting over again.

- new development can utilize the classification and organization scheme as well as reuse already existing models. Likewise, there may be a subset that represents a future state of the enterprise without regard to development.

The periodic table exists whether you are analyzing existing compounds to discern their composition (reverse engineering) or trying to create new compounds (new development) or whether you are just trying to do pure research on the elements themselves (learn about Enterprises and/or Enterprise Architecture).

- the models are as much about the operational side of the enterprise as the development side.

Implicit in the operating Enterprise are all of the primitive models ... it is only a question of whether you make them explicit or not. Making them explicit enables you to remove the erroneous assumptions (defects), reconcile dissonant perceptions (entropy) or engineer improvements (change). It is unlikely that you are going to be able to remove defects, reduce entropy or change the Enterprise appreciably without somehow or other building primitive models. Clearly, writing more code is not going to fix the problem.

#### TOTALLY INCORRECT ASSUMPTIONS ABOUT THE FRAMEWORK:

... higher rows of Zachman Framework differ from lower ones largely by abstraction, by suppressing detail - In other words rows are not different viewpoints (electrical, plumbing) (functional, non-functional) of same problem domain. ROWS - increasing greater level of detail as required by various observers or listeners to a story. Some observers require only a very high level of detail, while some require an in-depth, technical description of the story. ONCE AGAIN, THE ABOVE IS TOTALLY INCORRECT

#### HERE IS MY RESPONSE:

Level of detail is a function of the Cell, NOT the Column. In any one Cell, you could have a high level of detail (little detail), medium level of detail, or excruciating level of detail. Anyone who has EVER heard me talk has heard me say that ... "excruciating level of detail" is related to a Cell. What is making the Rows different is NOT more detail.

The Rows are different ... that is, different models occur in every Row within any one Column. It is just like in classic building architecture ... Row 2 is expressing the usage constraints of the end result as expressed by the "Owner" of the end result ... e.g. the Architects Drawings. Row 3 is expressing the constraints of the laws of nature as addressed by an Architect (or "Designer") ... e.g. the Architect's Plans.

(In the Enterprise domain, this would be the expression of the logical, systematic manner in which the conceptual requirements of the Owner might be effected.)

Row 4 expresses the constraints of the Builder in terms of the construction process and machine tool technologies being employed ... e.g. the Contractor's Plans. Row 5 expresses the "Sub-Contractors" tool specific constraints, out of context constraints. Row 1 is the "universe of discourse" relative to the analytical target ... in the case of Enterprises, out of the total universe of Things, Processes, Locations, Organizations, Cycles and Objectives, what is the subset relevant to this Enterprise that they are willing to put forth effort to manage.

Row 2, Row 3 and Row 4 are equivalent of Conceptual, Logical and Physical where Conceptual is what the Owners have in mind as to what their Enterprise is or what they want it to be. Logical is the systematic expression of how the Owner's intent could be logically realized. Physical is the technology constrained representations for implementation purposes.

Conceptual, Logical, Physical is the equivalent of the Work Breakdown Structure/Product Structure (Row 2), the Engineering Bill-of-Materials (Row 3) and the Manufacturing Engineering Bill-of-Materials (Row 4). They are DIFFERENT ... not increasing levels of detail because they are expressing different constraints.

For example in Column 1, Row 2 is a model of the actual Things of the Enterprise that the Owners care enough about to manage. Row 3 is a model of the logical representations of the things of the Enterprise which are DIFFERENT than the actual things because the "Designer" is trying to figure out how to keep track of the information (the filing system) needed to take inventories of all the Things of the Enterprise when the Owners need to know how many things there are or whether they are all present or not. Row 4 is a model of the technology manifestations of the Things which are DIFFERENT that the logical representations of Things and DIFFERENT from the Things themselves because the "Builder" is trying to figure out where to put the actual data about the things which is going to take up space on some device and you want to make sure the same data is in the same place every time you send an arm in to grab it off of the disk or whatever you are storing it on and you don't want to wait for long periods of time while looking for the data.

This is really important because you are trying to do different things with each different model. The models are RELATED to one another, but they are not the same model, just increasing levels of detail. Any one Row model has to support the intent of the next higher Row or else you are going to have a quality problem ... if the intent of any one Row is NOT supported by the next lower Row, when you get down to Row 6, the Row 6 implementation is not going to be "aligned" with the intent of the Owners of the Enterprise at Row 1/2.

Conceptual is like the card catalog in the library; Logical is like the Dewey Decimal System; and Physical is the shelf allocation for the books.

Similarly, Conceptual is like the subject matter index, Logical is the file folder identification and Physical is the location of the file folders in the file cabinets. Row 2 are the Architect's Drawings, Row 3 are the Architect's Plans and Row 4 are the Contractor's Plans. Row 2 are the Customer's requirements, Row 3 is the Engineering Design and Row 4 is the Manufacturing Engineering Design. How come those of us in the Data Processing Department are having so much trouble with this idea??? The Library people, the Document Management people, the Architecture and Construction people and the Engineering and Manufacturing people don't seem to have any trouble with this at all!!

SOME OF MY ANSWERS TO SEVERAL QUESTIONS THAT WOULD BE TOO HARD TO FIND AND ARTICULATE. HOPEFULLY THE QUESTIONS WILL BE APPARENT IN MY ANSWERS:

First, the Framework is a two dimensional classification scheme for descriptive representations of an Enterprise. I discovered it by looking at the structure of descriptive representations of airplanes ... and we examined buildings, automobiles, battleships and super computers as well.

I just applied the same structure of product description artifacts to Enterprise descriptions. It has nothing to do with systems analysis and design techniques or network architectures other than the fact that some of the descriptive representations of the Enterprise may be produced over the process of employing stored programming devices and electronic media for implementation purposes. I might observe that it is equally conceivable that descriptive representations could prove useful when using pencils, paper and file cabinets as the implementation technology as well. I also might observe that those are your two implementation options ... pencils, paper and file cabinets or stored programming devices and electronic media. The same Framework for descriptive representations could be used to describe the material handling and people moving technologies but those technologies tend to be beyond our area of interest. The technology constraints being employed in the implementation of the Enterprise, whatever they are, simply affect the content of the Cell models at Row 4 ... not the Framework itself.

Second, the Framework is a semantic structure and therefore does not imply anything about process ... process is a methodology or tool issue ... and a meta issue. Which Cells or Cells, or which "sliver" of which Cells in what sequence they are produced is a function of the value system of the author of the methodology or tool. The Framework is inert relative to methods/tools. The reason why UML might not address Rows 1 or 2 models is probably because UML's origin was Row 5 and subsequently, Row 4. I have a friend that quotes Grady Booch as saying "we know how to do design, we don't know how to do analysis." I cannot validate the quotation, but assuming it is somewhat accurate, I would interpret that to mean the object folks know how to do Row 4, not Row 3.

People tell me that there is some problem using UML to do Row 2 and I suspect that the underlying problem is the meta-model of UML does not support the differentiation between Enterprise things (Row 2) and logical representations of Enterprise things (Row 3).

Regarding "removing the wall between Data and Process" ... that also likely stems from the origin of the object community. I suspect that their overriding value proposition was implementation ... not architecture. For implementation you need composite models (for example data and process together) whereas for architecture, you want primitives (data and process separate). Ultimately, you want both composites and primitives because you want to create the composites from the primitives. If you are only creating composites on an ad hoc basis, then you have point in time solutions relative to a specific implementation ... that's the heart of the "legacy" problem. The end object of architecture is to create an inventory of reusable assets defined relative to the Enterprise (not relative to a single implementation) from which a wide variety of composites (objects) could be assembled to order ... NOT created to order.

Regarding CMM ... clearly CMM is a process issue by it's very definition. I think the question they are addressing is, how much formality and discipline exists in the application development process. I suppose you could get some insight into the amount of formality and discipline by assessing which slivers of which Cells of the Framework does the application development process formally produce and manage. But, there is probably an even more profound question that could be exercised and that has to do with composites and primitives. If the application development process produces composites, it is likely a typical "job shop". It is making custom products ... applications. No engineering or manufacturing is being done until they get the order. On the other hand, if the methodology is producing primitives, then it is likely that they are changing the fundamental concept of I/S to "mass customization." They are creating architectural primitives relative to the Enterprise before they get the order for an application so that when the order arrives, the time for implementation is reduced to only the time required to assemble the composite from the primitives ... the engineering has already been done.

Clearly, if you want to reduce the "time-to-market" for systems implementations and at the same time accommodate enormously increased complexity in the Enterprise (for example, CRM), you have to have something in inventory that has been engineered to be assembled into more than one implementation before you get the order. Clearly, CMM does not address that issue at all. It is merely measuring the degree of formality and discipline in the process whatever process is being employed. On the other hand, if there is little formality in the process, the likelihood of realizing architectural approaches (mass customization) for the Enterprise I would judge to be low to zero.

I am very concerned about the gross misconceptions so many people have relative to my Framework specifically and to Enterprise Architecture in general. I have been doing seminars regularly in the U.S., Europe, Asia, and Africa for over 10 years. I have been attempting to write more extensively for the last several years and have written about 20 major articles during that time. There are some highly knowledgeable Framework people that I have worked with for over 20 years that are now doing extensive, more in depth seminars about Enterprise Architecture and my Framework than I could possibly do. I have finally finished writing "The Book" which we expect to have in electronic production soon. I am not too sure what else I can do in the face of a growing body of "experts" on the Zachman Framework that have never gone to my seminars nor read any of my materials.